VIS AND ANALYSIS R&D FOR LARGE-SCALE DATA

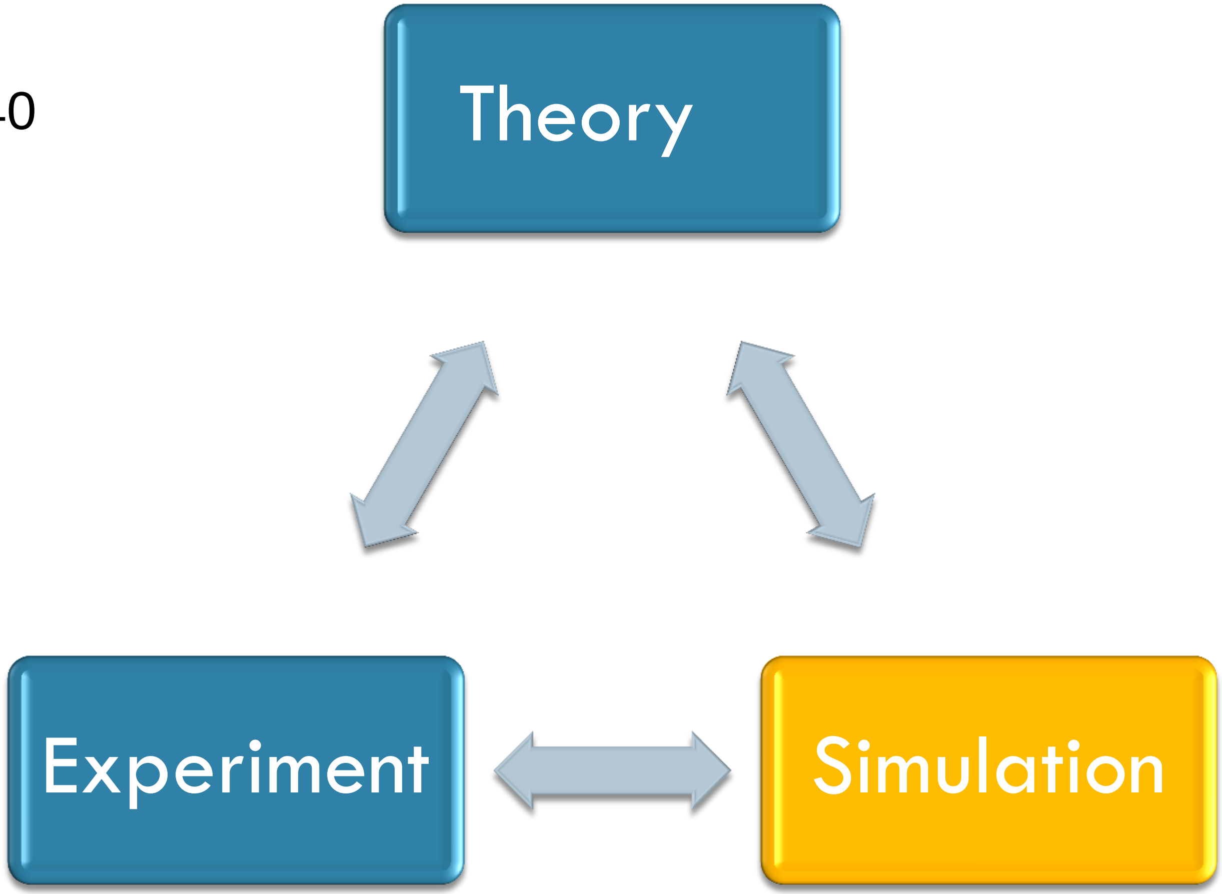
Jon Woodring

et al.: Jim Ahrens, John Patchett, Chris Sewell, Li-Ta Lo, Chris Mitchell, Pat Fasel, Joanne Wendelberger, Kary Myers, Curt Canada, Rick Knight, Hilary Abhold

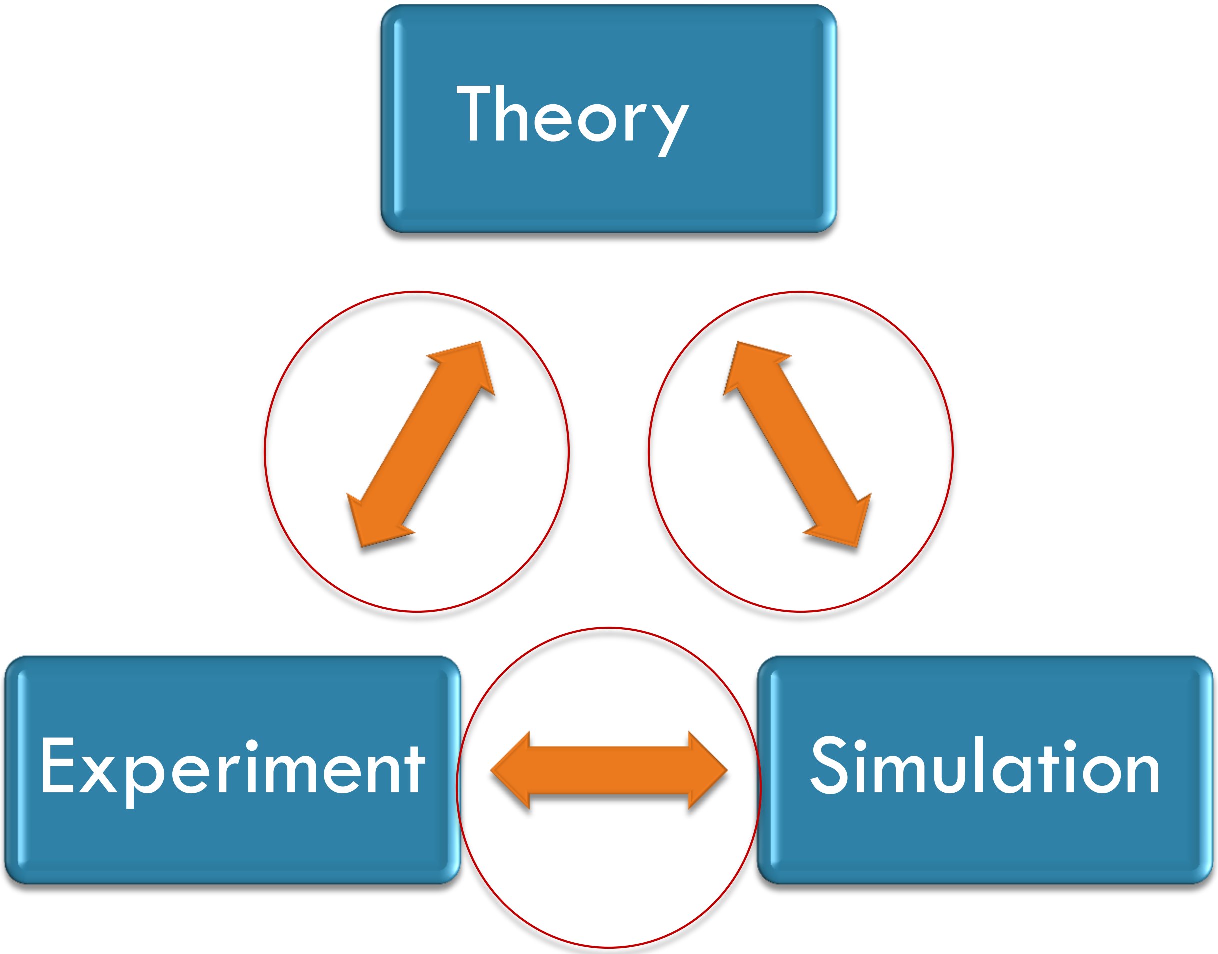
Salishan 2013 – LA-UR-13-21885

Science!

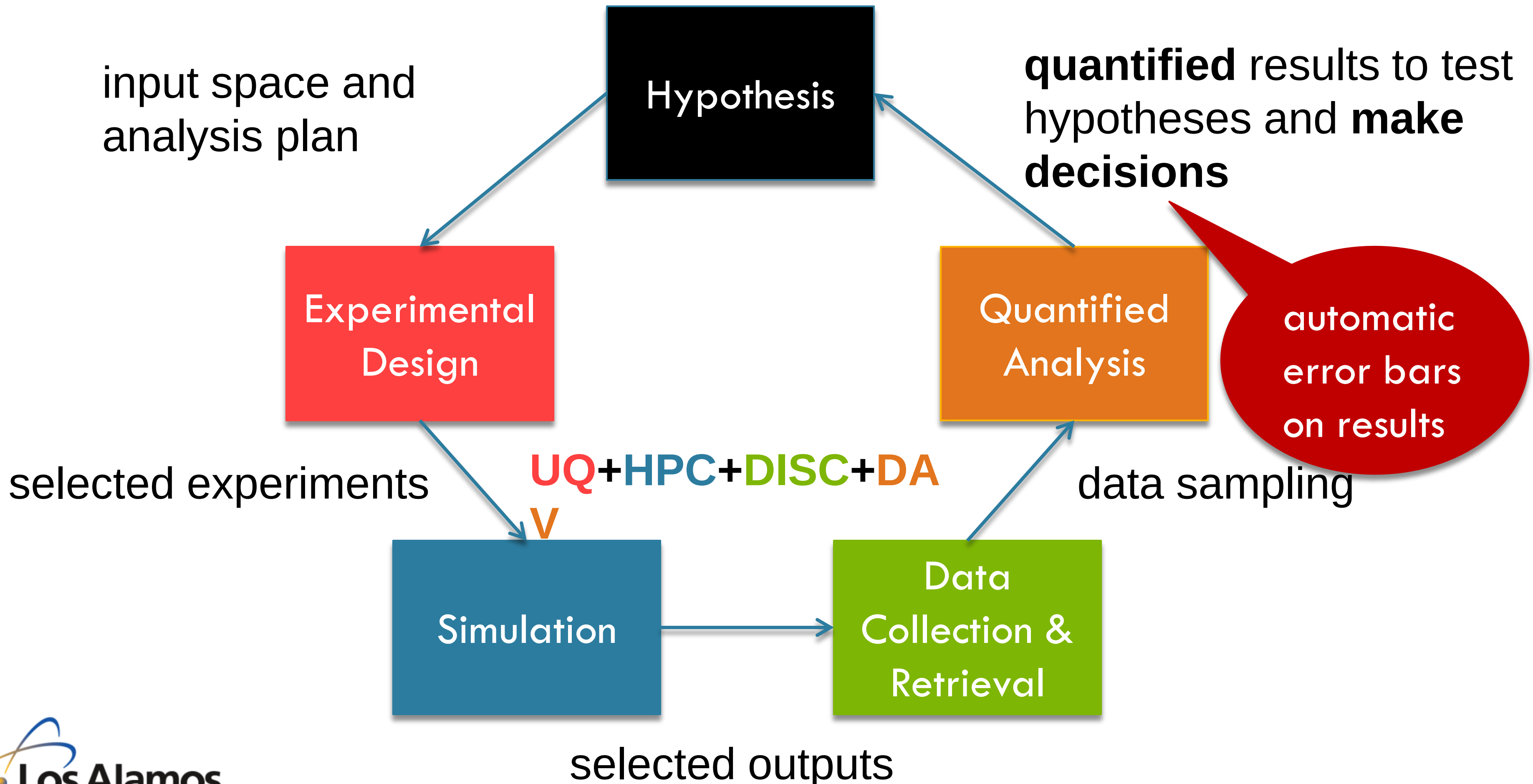
After ~1940



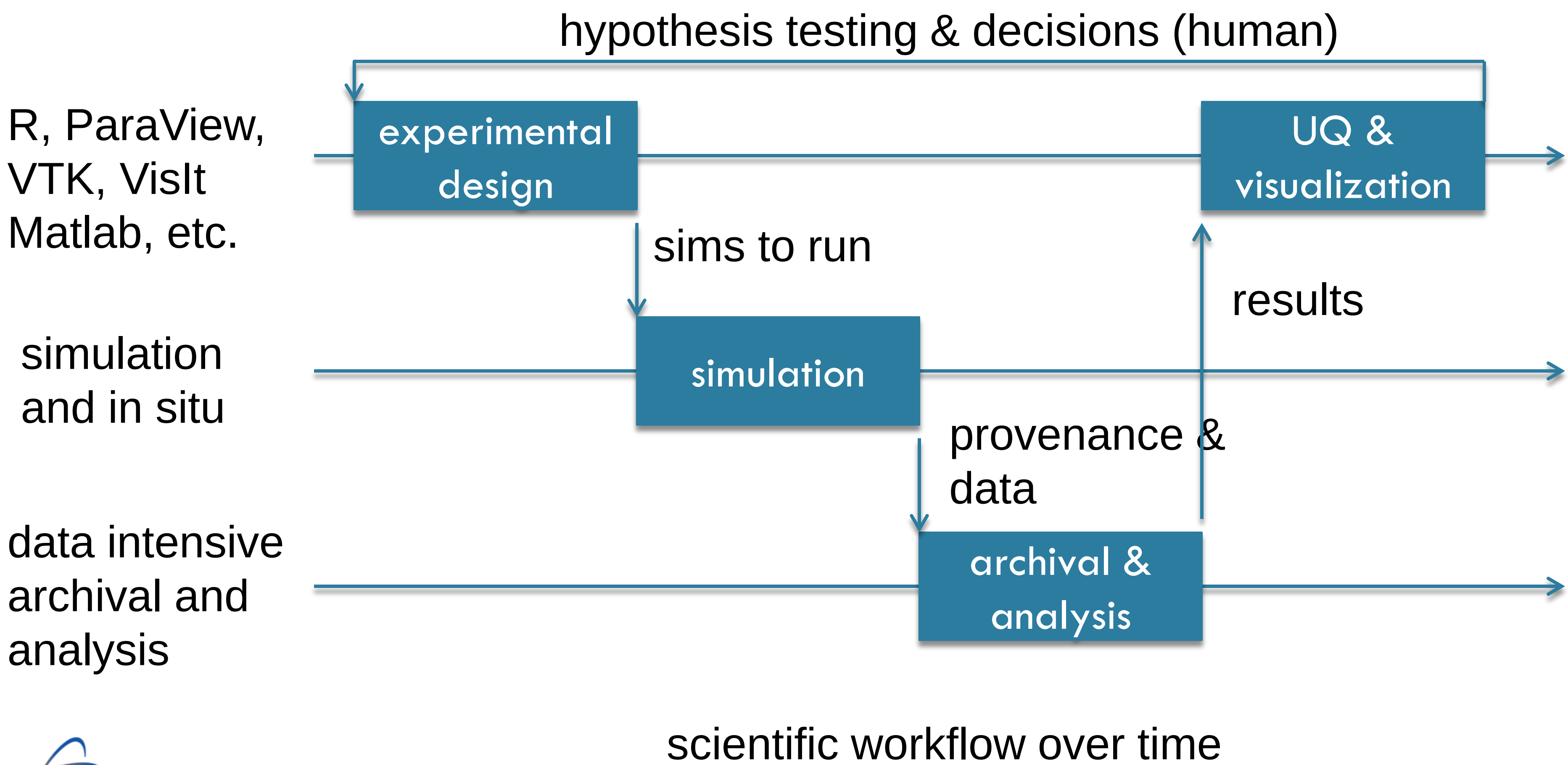
Modern Data Driven Science Research
LANL Data Science at Scale Team



My Vision for **HPC Simulation** Science
"Error Bars on Everything!"



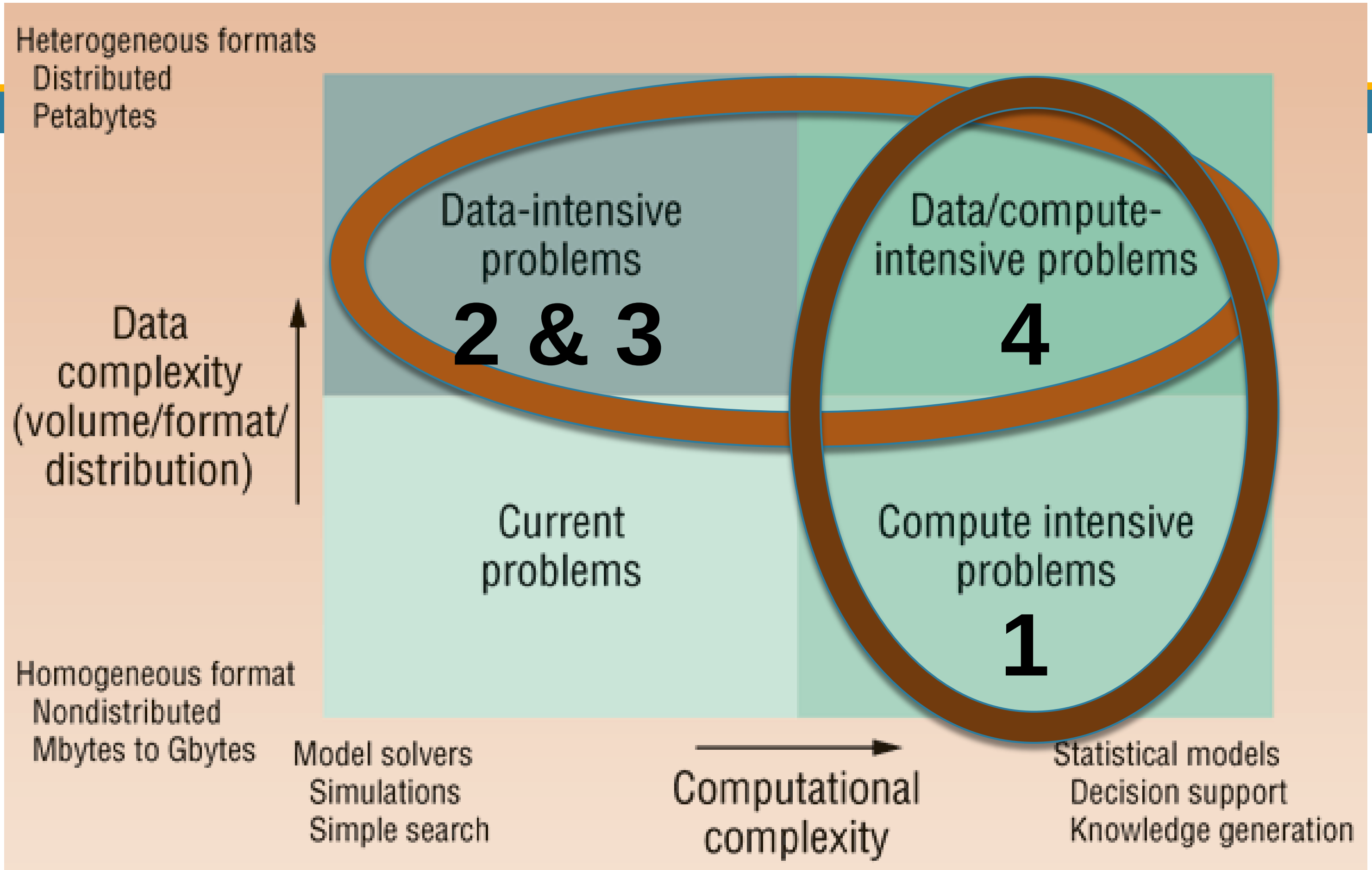
Vision for HPC Scientific Workflow



Some Current Research to Get There

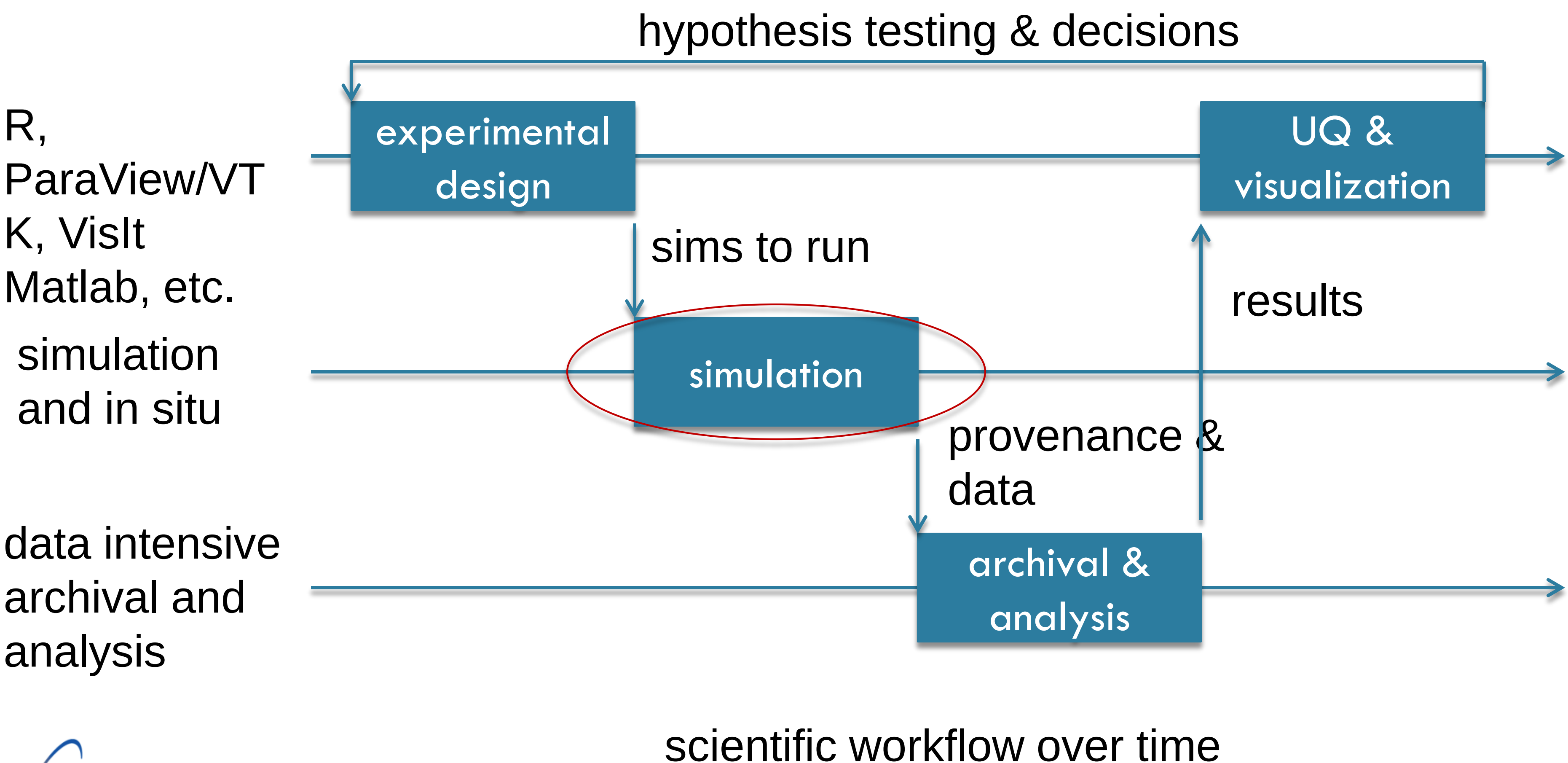
- 1) Mesh sharing for no-copy (shallow) run-time translation
  - *Exascale*
  - Jon Woodring, Tim Tautges (ANL), Tom Peterka (ANL), Venkat Vishwanath (ANL), Berk Geveci (Kitware)
- 2) Data selection and 3) Quantified analysis for managing simulation data
  - *Data intensive*
  - Jon Woodring, Kary Myers, Joanne Wendelberger, Jim Ahrens, Chris Brislawn, Sue Mniszewski
- 4) Co-design of burst buffers for analysis use cases
  - *Exascale and data intensive*
  - Jon Woodring, Chris Mitchell, Aaron Torres, Mat Maltrud, Rick Knight

Exascale and Data Intensive R&D



Ian Gorton, Paul Greenfield, Alex Szalay, Roy Williams, "Data-Intensive Computing in the 21st Century," Computer, pp. 30-32, April, 2008.

1) Mesh sharing for no-copy (shallow) run-time translation

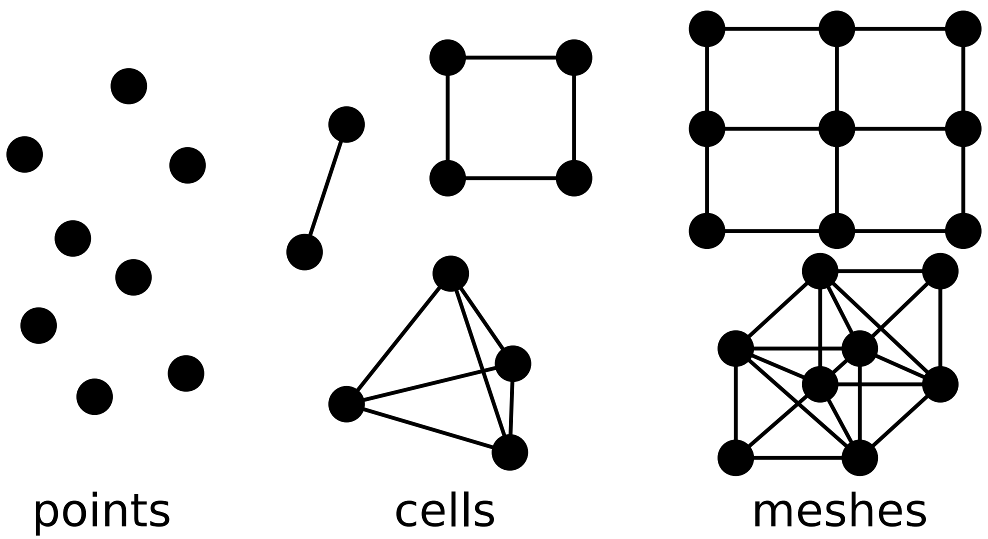


Exascale Memory Constraints Share via On Demand Translation

- **Memory constrained per process – going down per core**
  - Total memory going up, but not as fast as cores
- DOE codes share data (memory coupled):
  - Multi-physics, in situ, IO libraries, etc.
  - Usually duplicate data from one code to the other, wastes memory – “deep copy” of data
- Share pointers (references) to the data structures?
  - Fragile and prone to error
- **Do on-demand, fine-grained data translation**
  - i.e., do translation in small-chunk “get datum” methods
  - Translation shim code (sometimes called a “thunk”)
- **Does it scale? How much memory do we save?**

Mesh-based Data (MOAB and VTK) Fine-grained, on demand conversion

- Many DOE codes use a mesh-based data model (for finite element methods, vis and analysis, etc.), with different interfaces and implementations

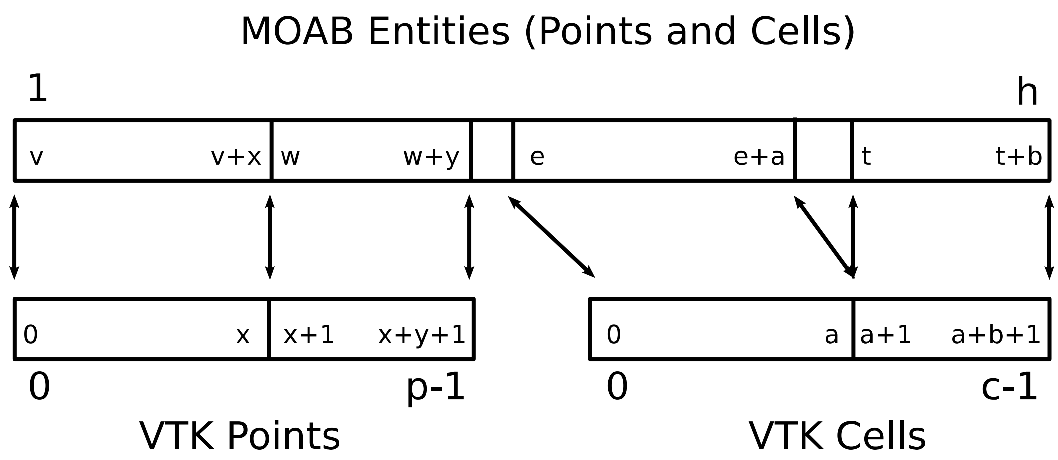


Unstructured Mesh Data to Convert at Run-time:

- Point list (coordinates)
- Cell list (cell types, point connectivities)
- Attribute data (point and cell data)
- Field data (data set information: time step, block number, etc.)

Most of the work is in id translation during a "get point" or "get cell"

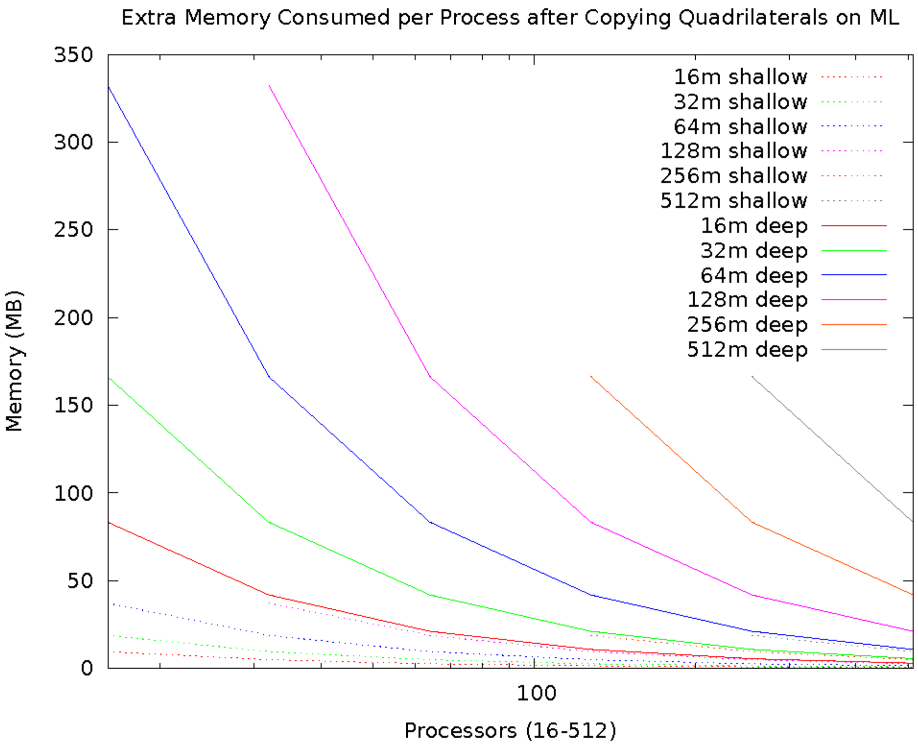
- VTK and MOAB address points and cells in different ways
  - VTK dense, 0-indexed, two namespaces (points and cells)
  - MOAB dense and sparse, one namespace (all entities)

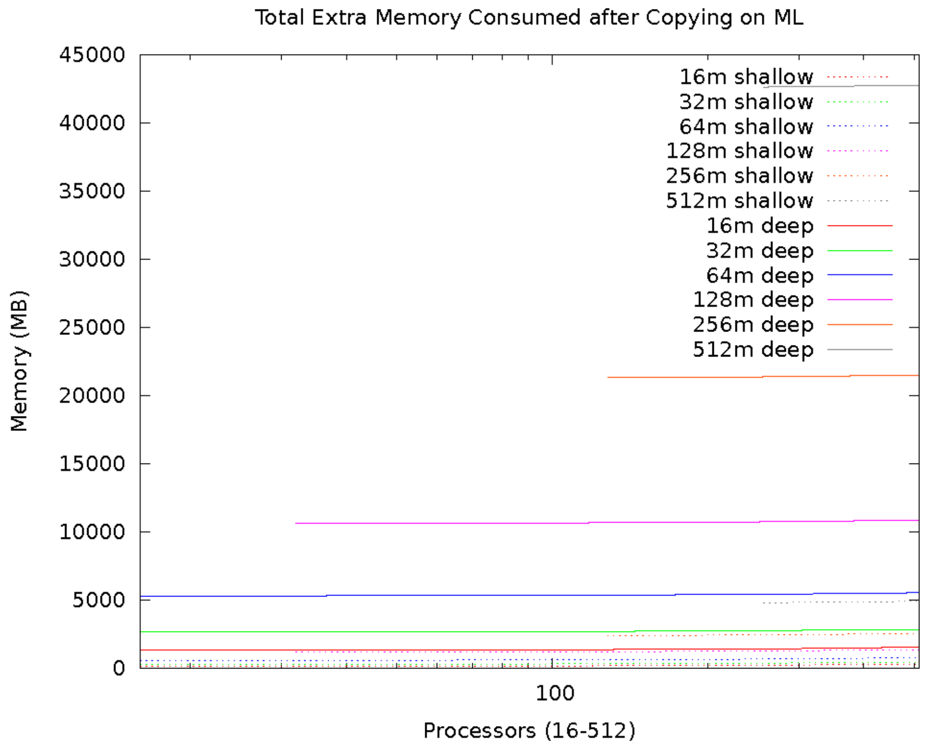


Done with a map and some math:
dest = M.lower_bound(src) + src

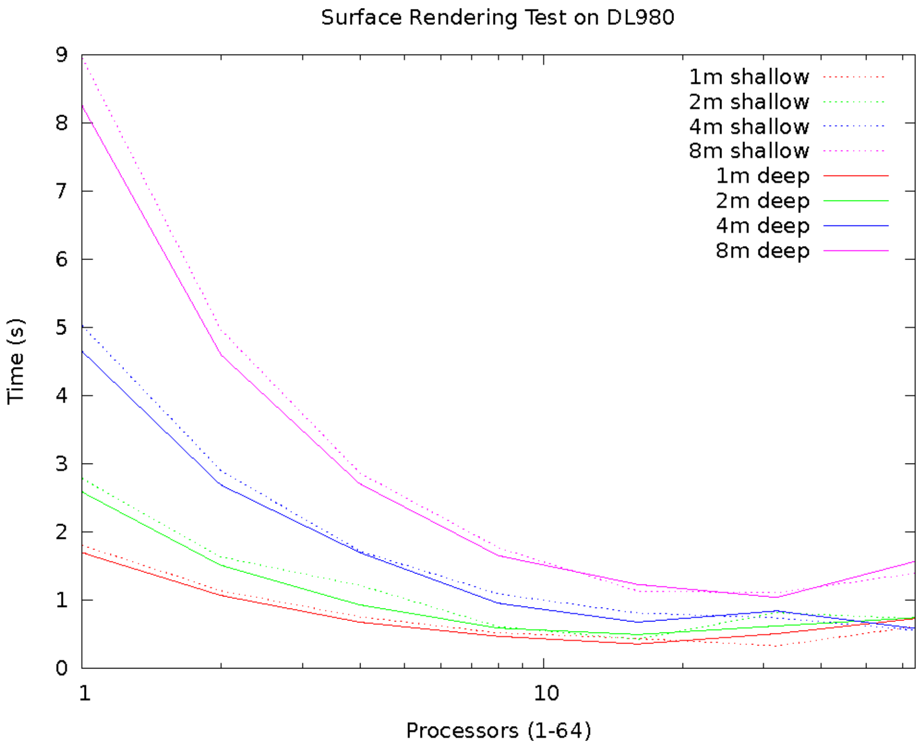
When we "get cell" from VTK, it gets converted into a "get cell" for MOAB at run-time

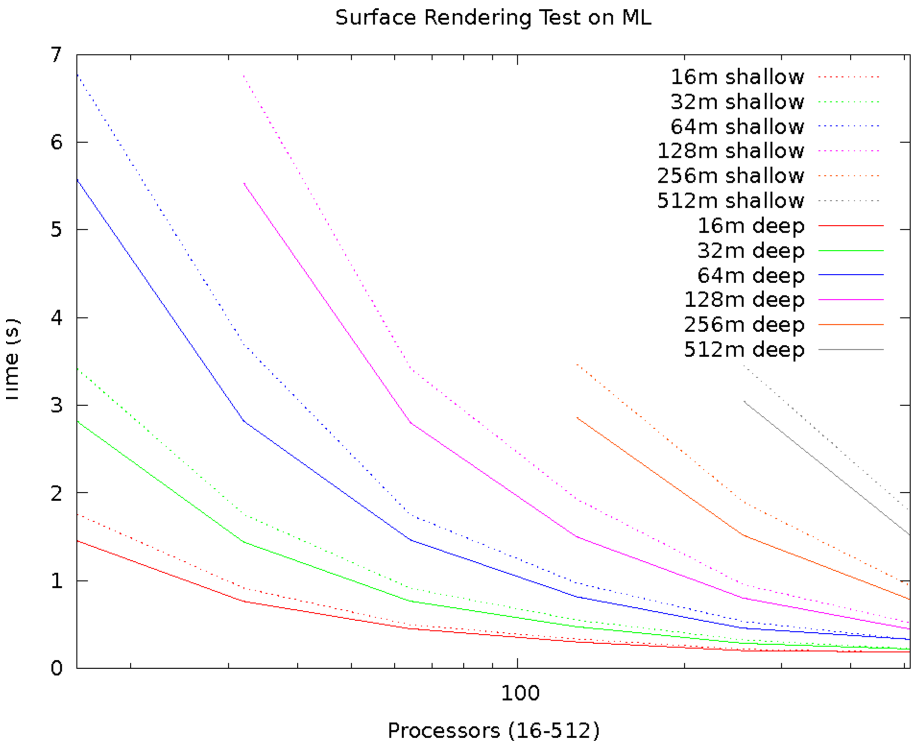
Copy Memory Performance (Moonlight)
Deep Copy costs x9 as much memory





Render Time
worst: DL980 x1.05, ML x1.15 slower

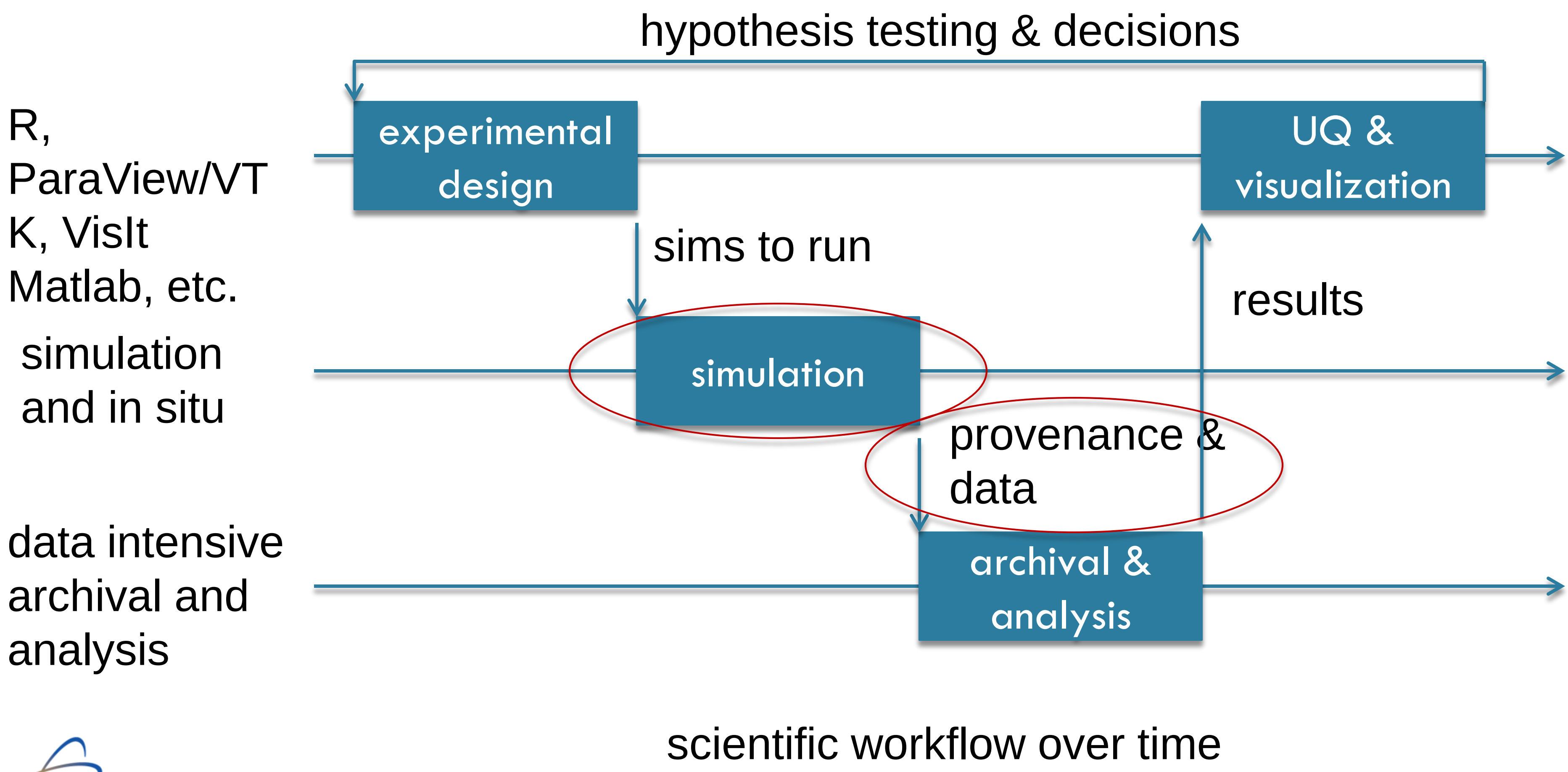




Promising Results: Time overhead is not too bad, memory savings is good

- Probably suitable for one shot operations: in situ, IO; maybe for suitable for compute heavy: clip, contour, render testing seems to indicate so
- Could be optimized – no optimization currently
- Need to try this with row-oriented (array of structs) meshes and IO libraries
- Eternal struggle of code reuse vs. peak performance priorities
  - Makes it easy to share data, trading compute speed

2) Intelligent Data selection for managing simulation data



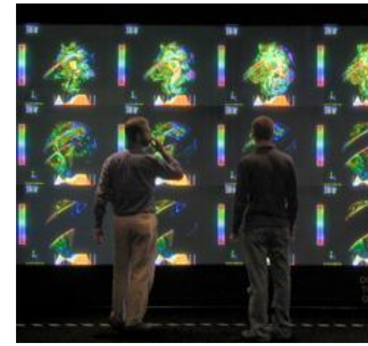
Data Glut for HPC **and** Scientist
We need data reduction

- I/O is the bottleneck for both the large-scale (HPC) simulation and analysis tool (I/O bound)
  - Most of post-processed or batch visualizations and analyses time is spent in I/O just loading the data
  - Simulations are already throwing away most of it
- Cognitive bandwidth bottleneck (human bound)
  - The data sizes have already exceeded the human cognitive capacity to be able to look at all of the raw data
- In situ is one way to tackle it, but at a price, losing the "discovery" from human-in-the-loop

Interactive Analysis vs. In Situ Analysis
How do we keep the "discovery"?

sparse
raw data

"dynamic"
discovery

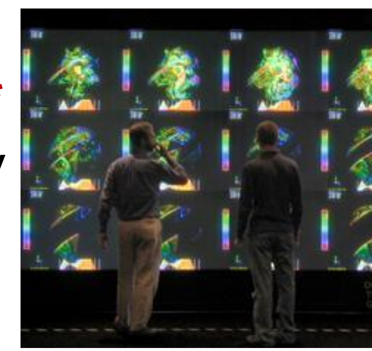


interactivity

raw data

in situ
products*

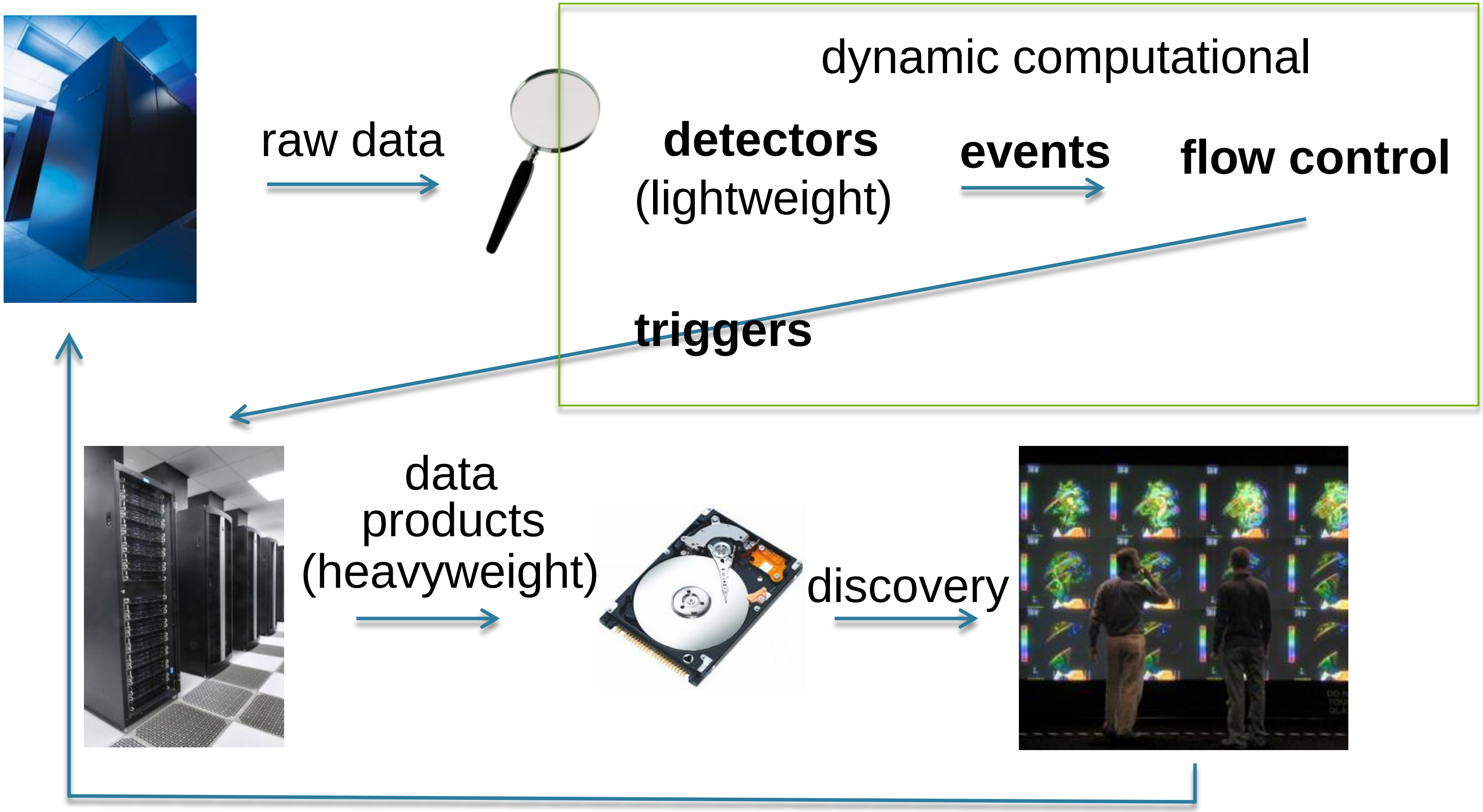
"static"*
discovery



refine (if missed something, rerun the sim)

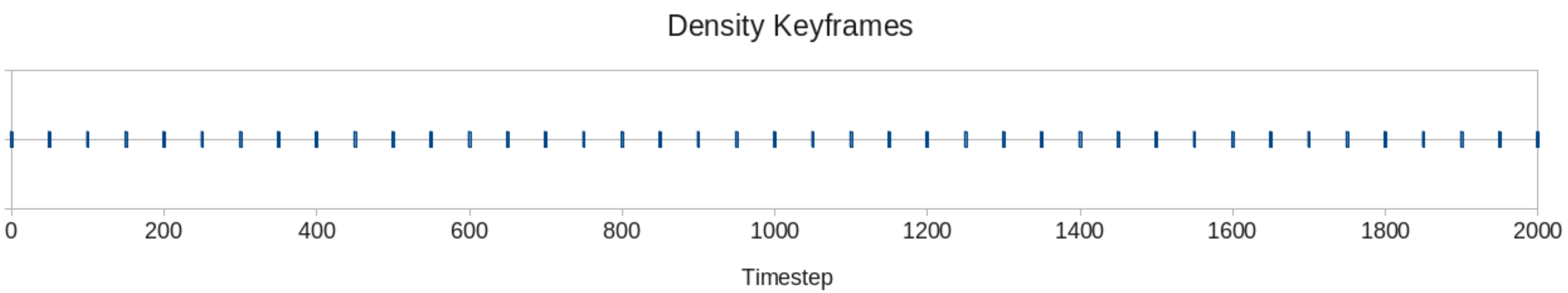
*some in situ output allows interaction: explorable images, etc.

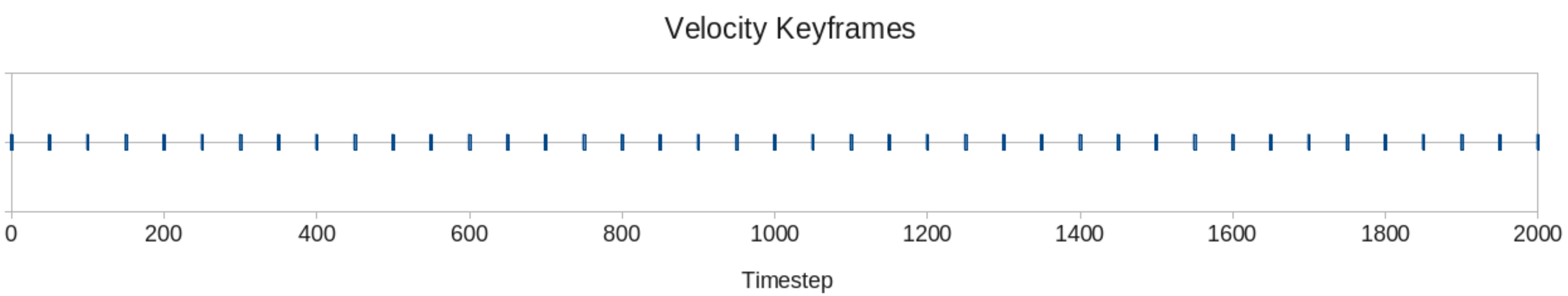
Adaptive In Situ: Feature selection
Intersecting Interactive & In Situ



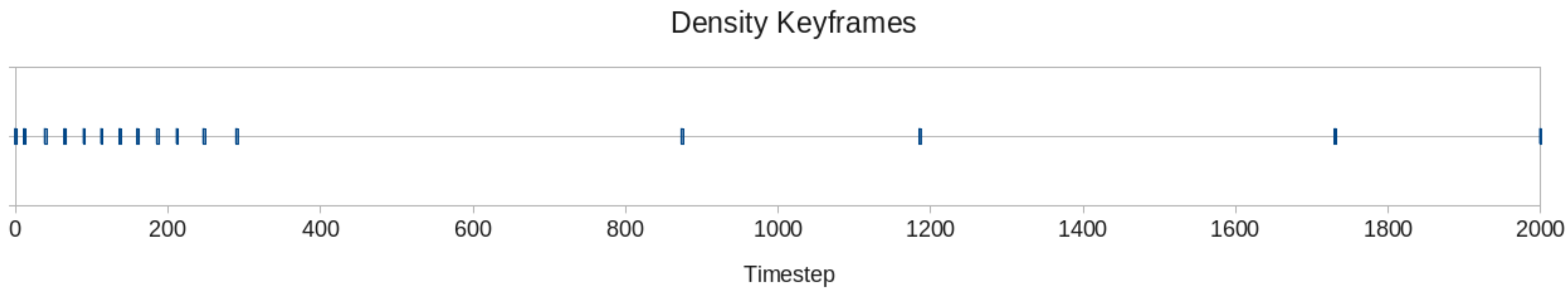
re-run the sim (less often than normal in situ)

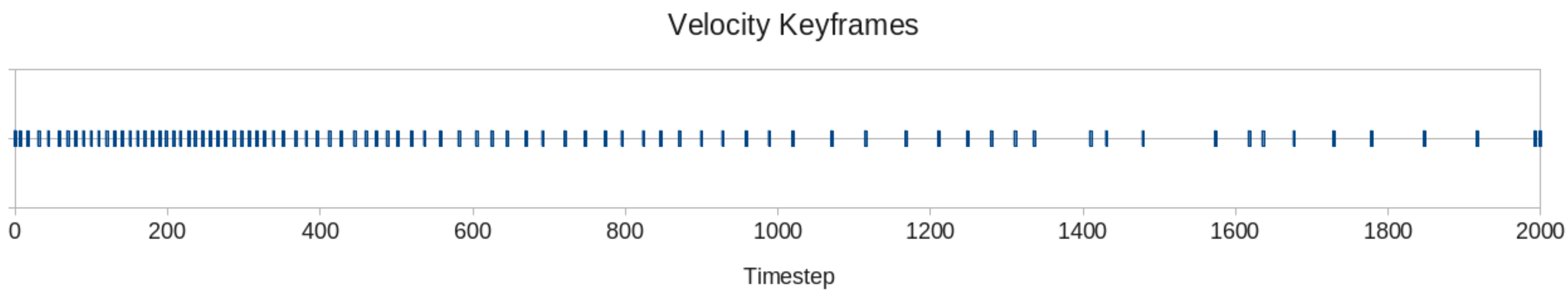
Typical Uniform Output
No knowledge is applied to selection



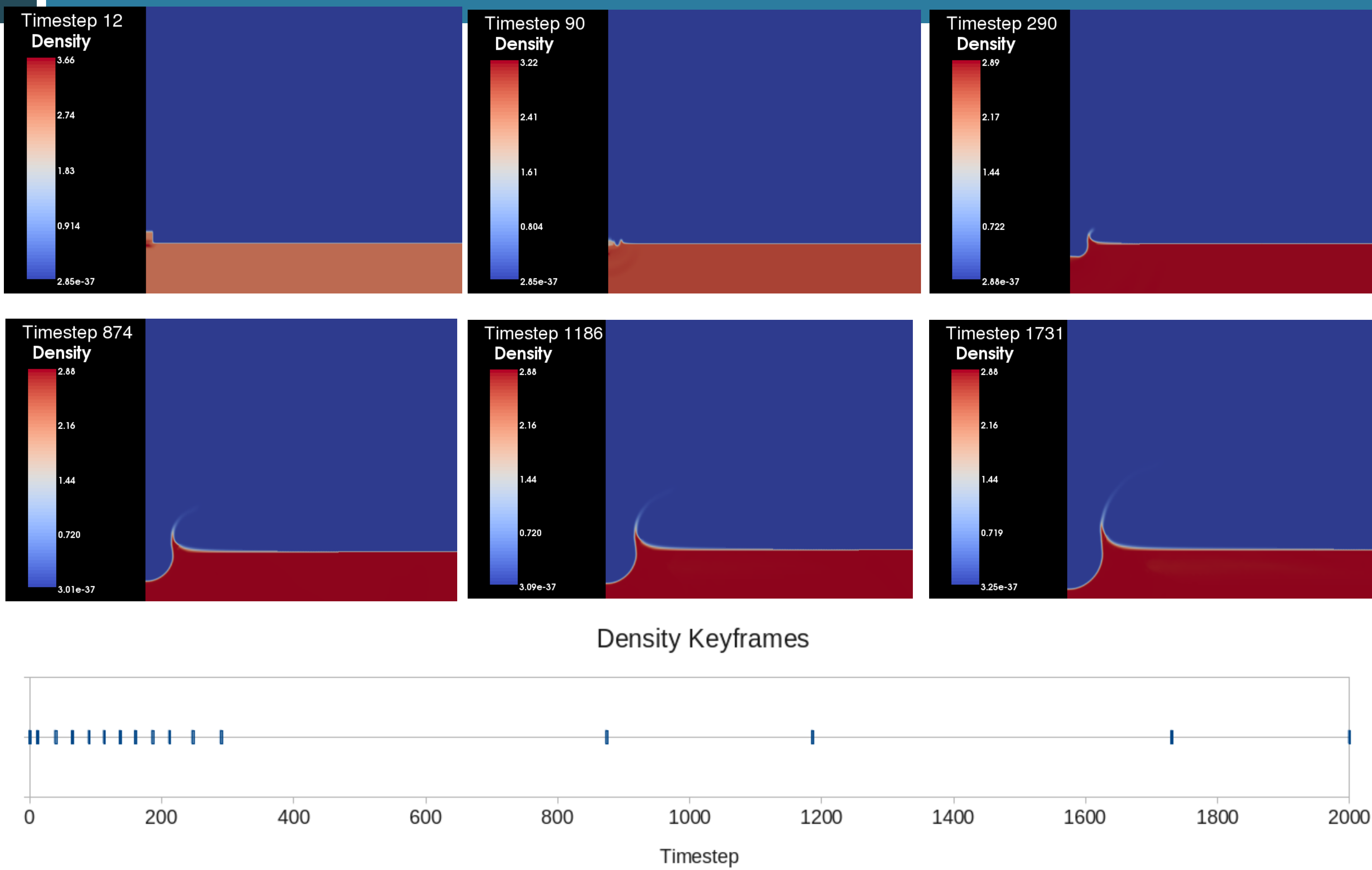


Statistically Driven Data Selection
Optimize Bandwidth (Human and HPC)





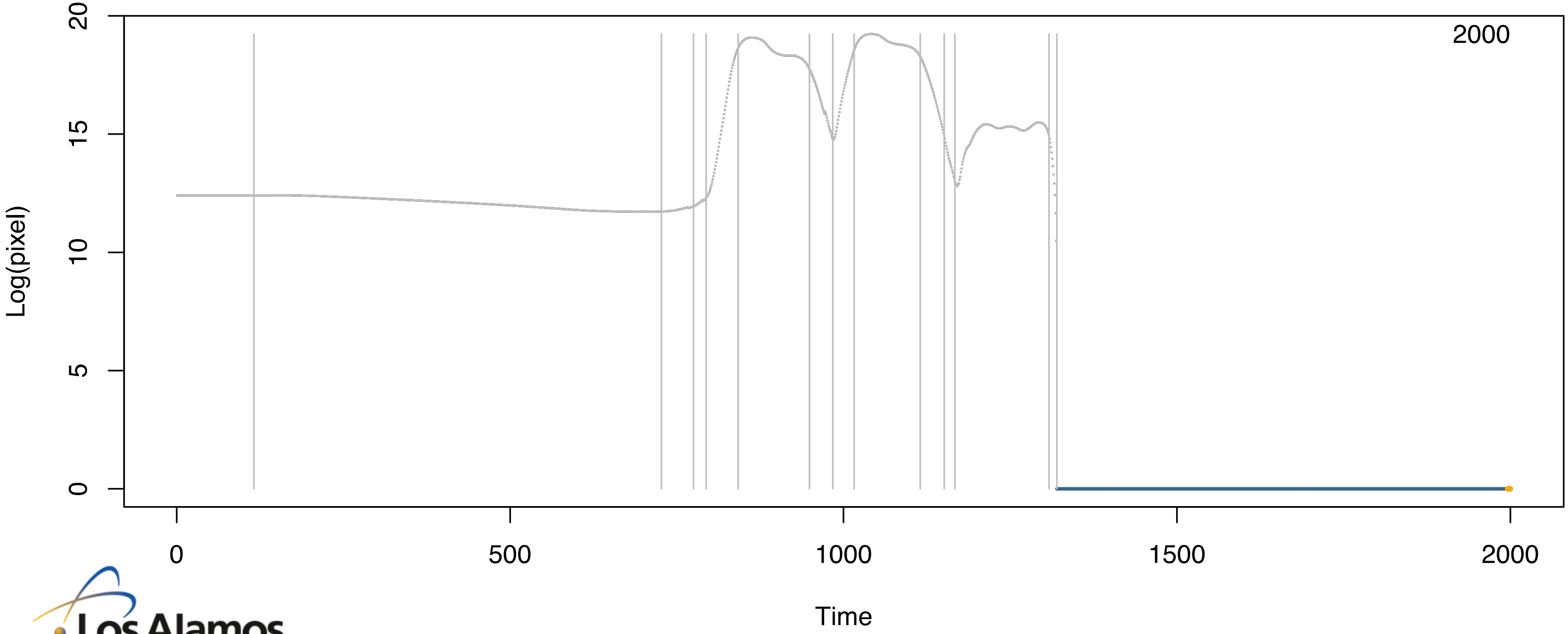
Selected Frames on Density in XRage
Uses histogram metrics to compare



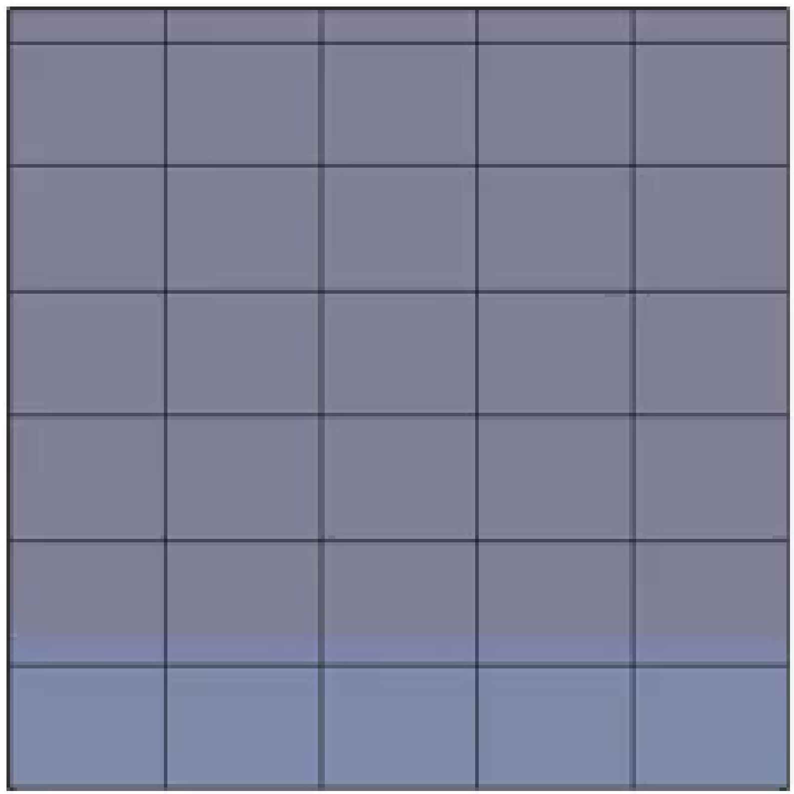
Selection of Time Steps from Simulation Using f-test for linear time prediction

The choice of α governs the triggering mechanism of the f-test.

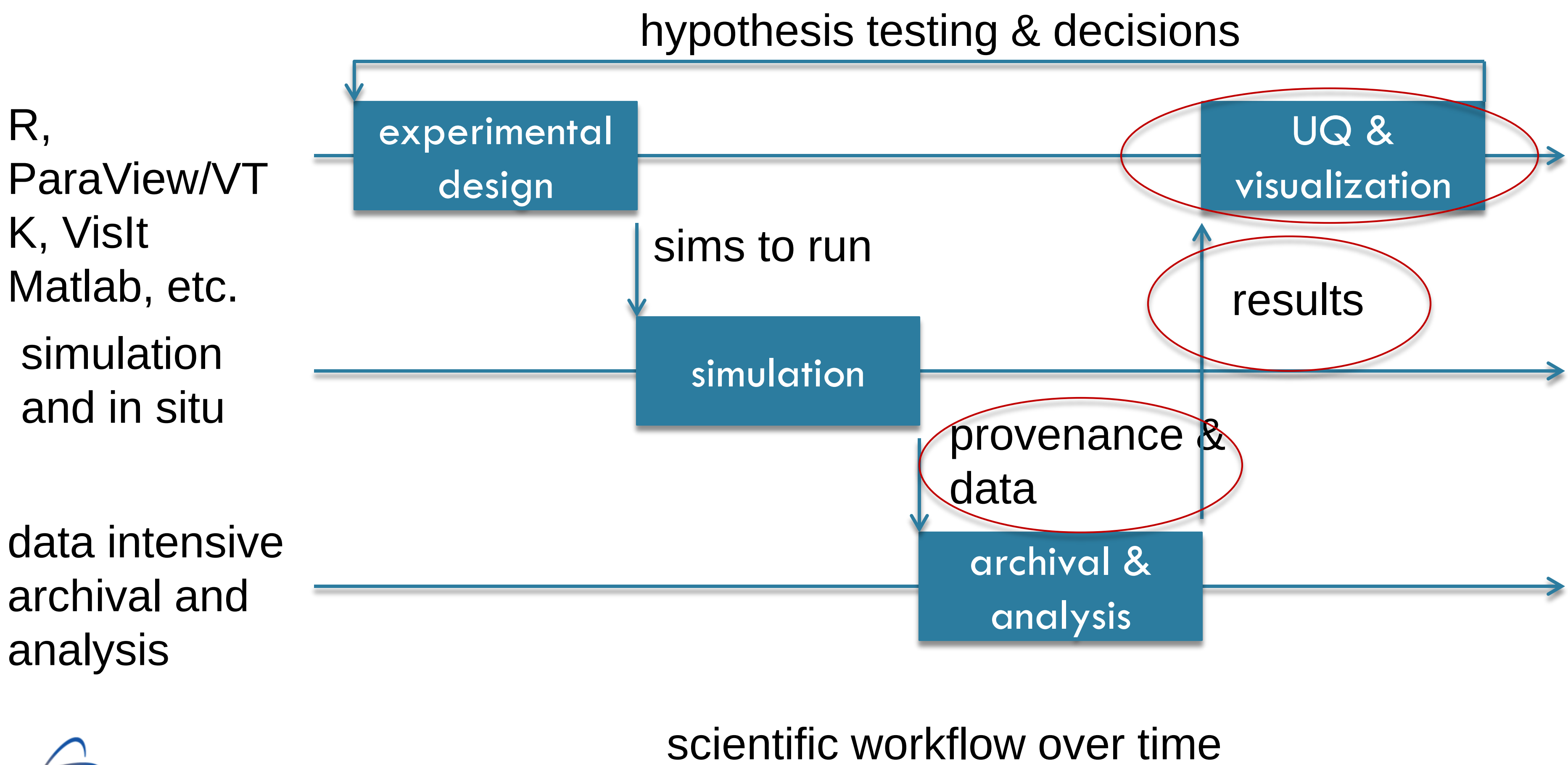
14 distinct regions selected with $\alpha = 5 \times 10^{-6}$:



Demonstration with LCROSS satellite impact simulation in Xrage with f-test

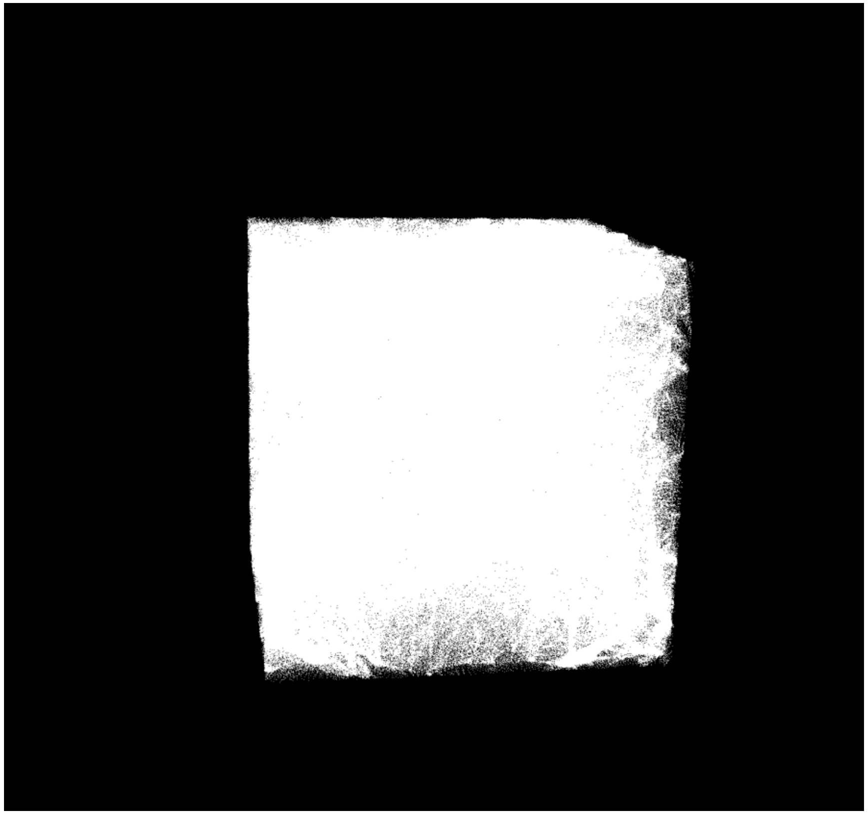


3) Quantified Visualization and Analysis for Managing Data Reduction

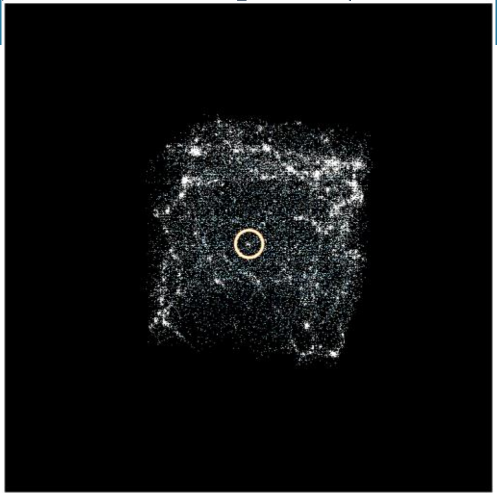


Simple Example of Too Many Data We can reduce it, but what is lost?

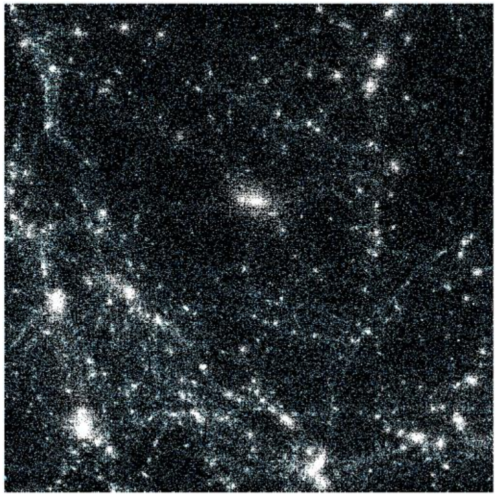
- The data on the right aren't even "large" – there are "only" 256^3 (16 million points) on a megapixel display
- Even for visualization, we often need to reduce the data, in addition to reducing data for storage bottlenecks (compression, sampling, in situ, etc.)
- **But, what do you lose?**



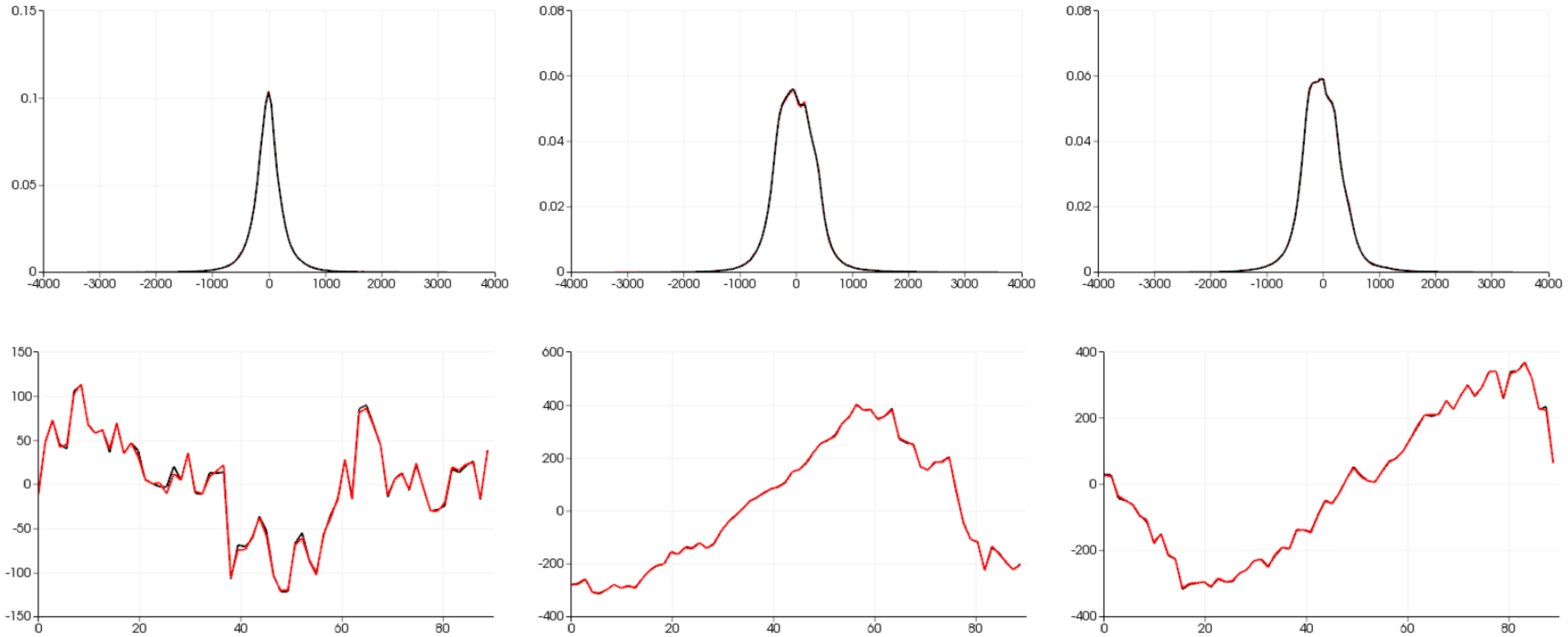
Quantify (Provenance Error Log) the Differences in the Data (Always!)



- Always measure the differences between reduced data and original resolution data before storing a reduction (the data aren't "lost" yet)
- Comparing provides for a bounding metric on the stored data – a provenance of the transformation
- Record the differences at all stages of analysis and reduction to quantify the differences in the scientific workflow

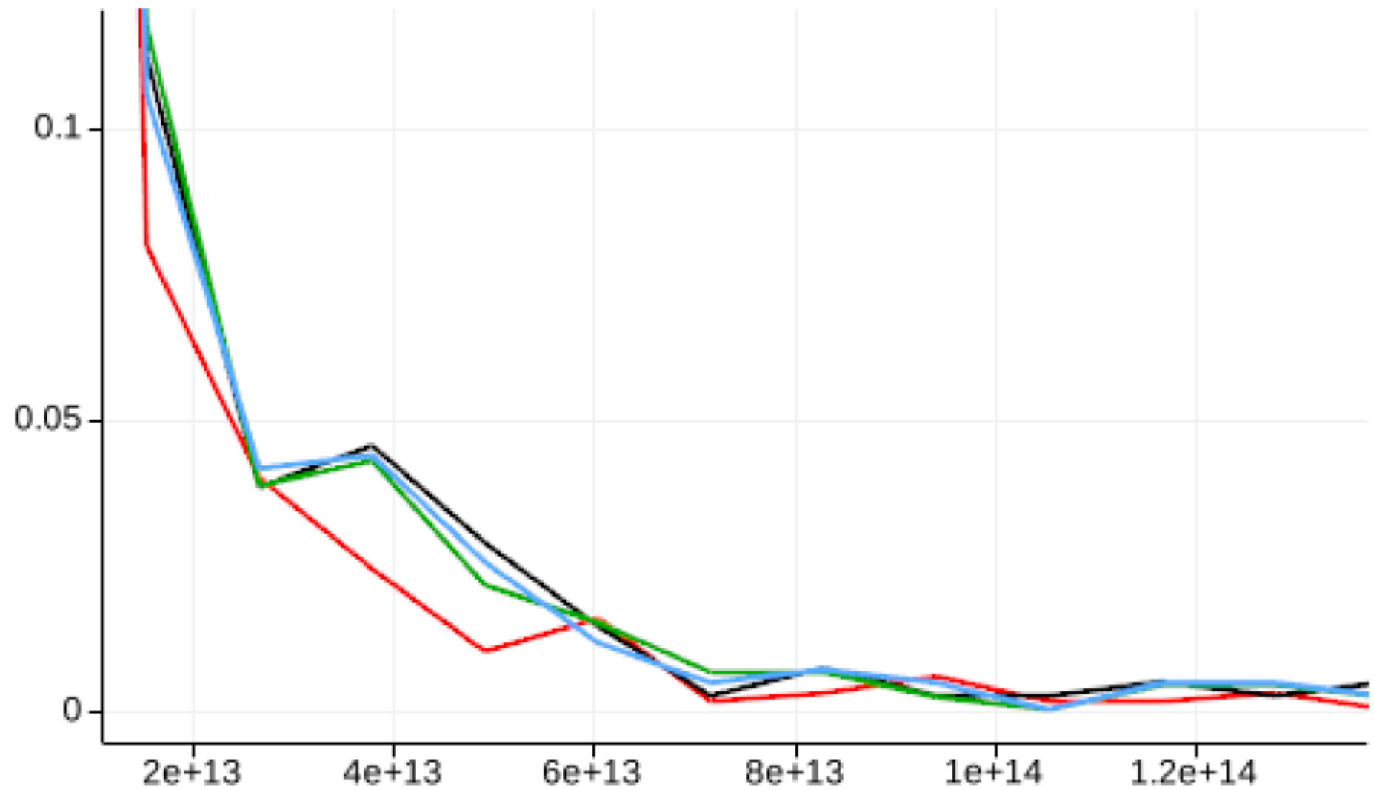


Provenance Error Log Comparing a 0.19% Sample to Full Resolution Data

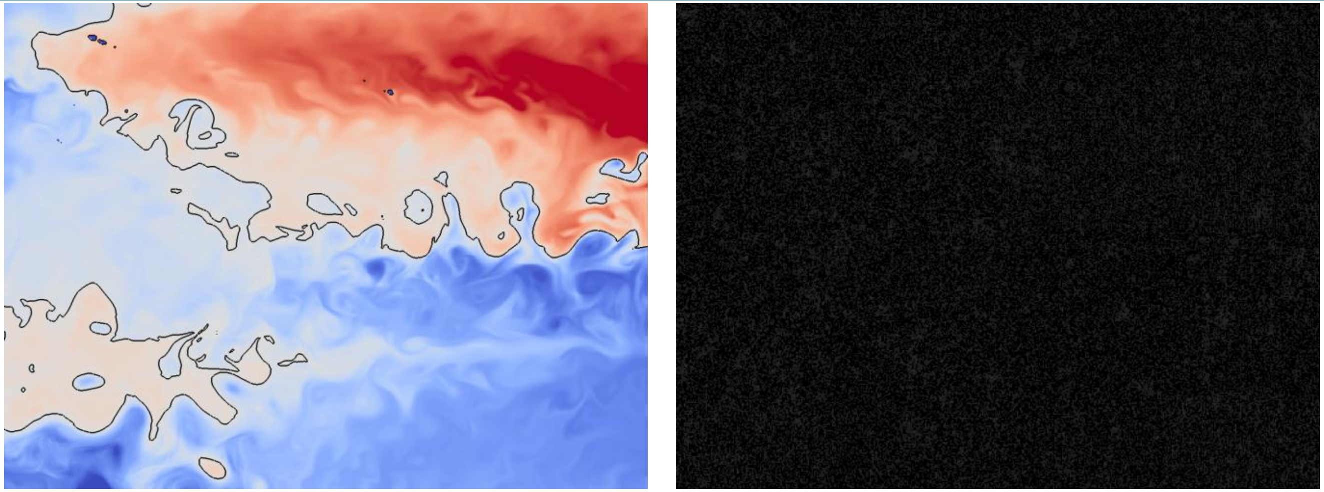


Los Alamos NATIONAL LABORATORY EST. 1943

Provenance of Comparing Halo Analysis on Different Data Reductions

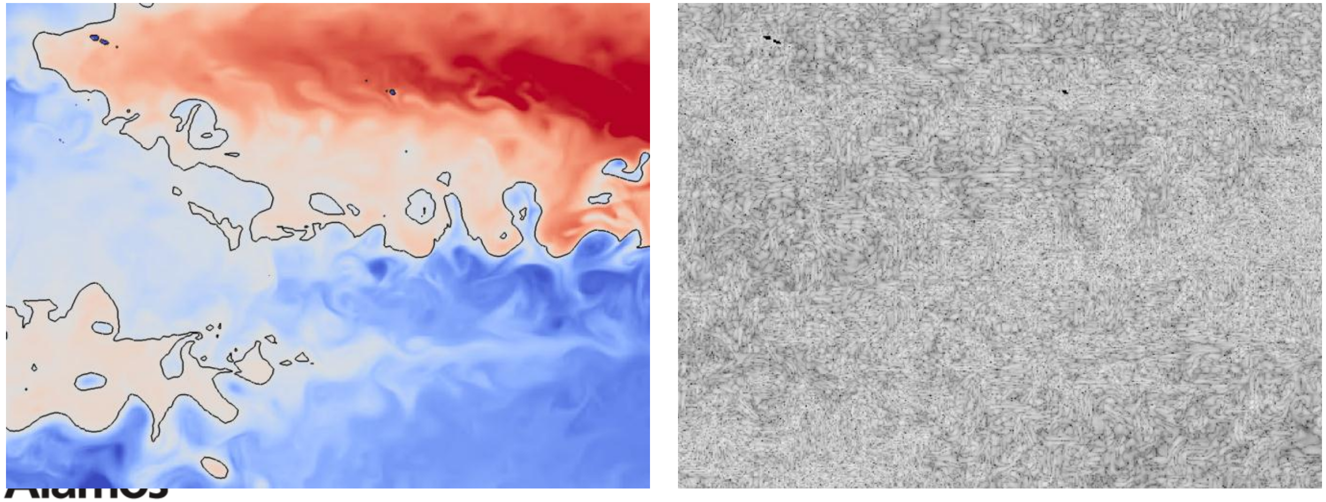


Point-wise Differences, Max error, and Isocontour Error Provenance



8bpp

+/-1.49012e-08

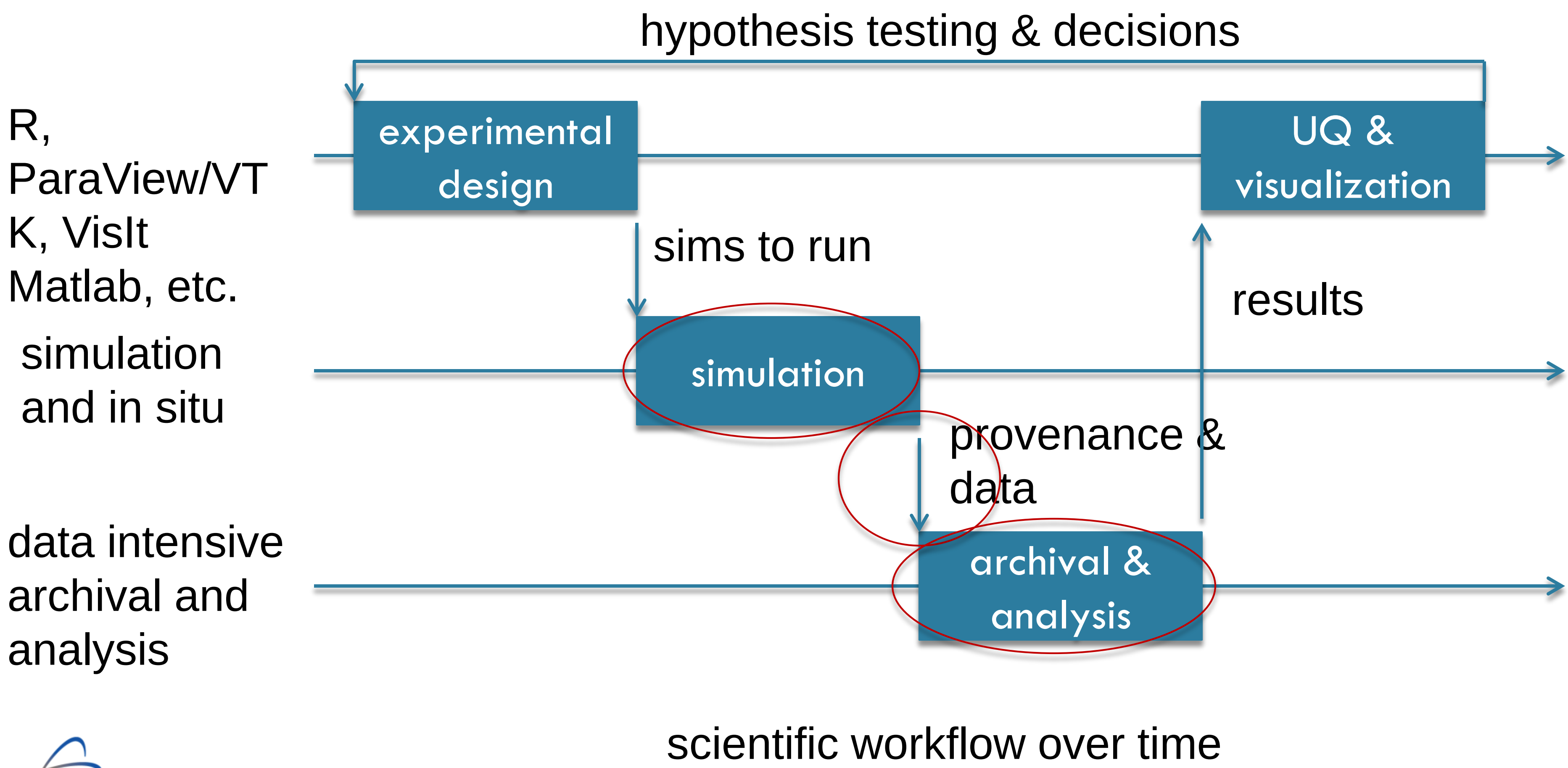


0.5 bpp

+/-8.58903e-05

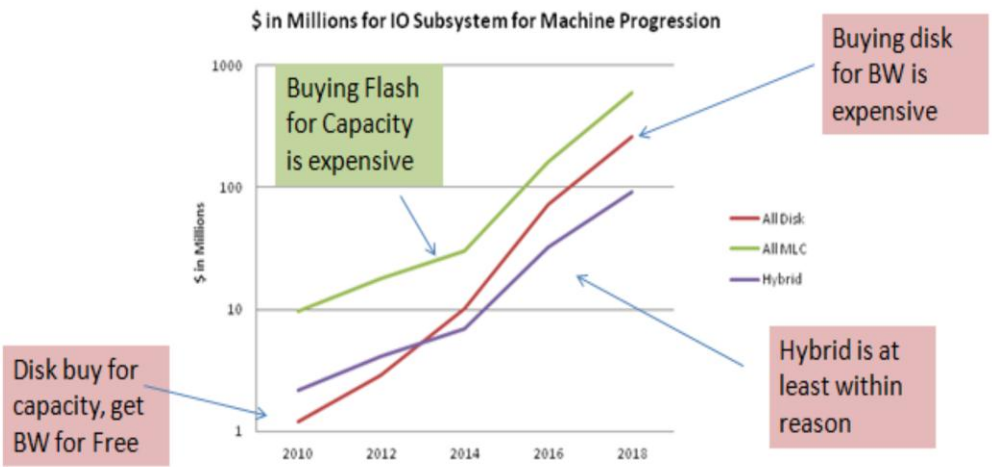
Maximum Error vs. Bit Rate after Compression Error Provenance

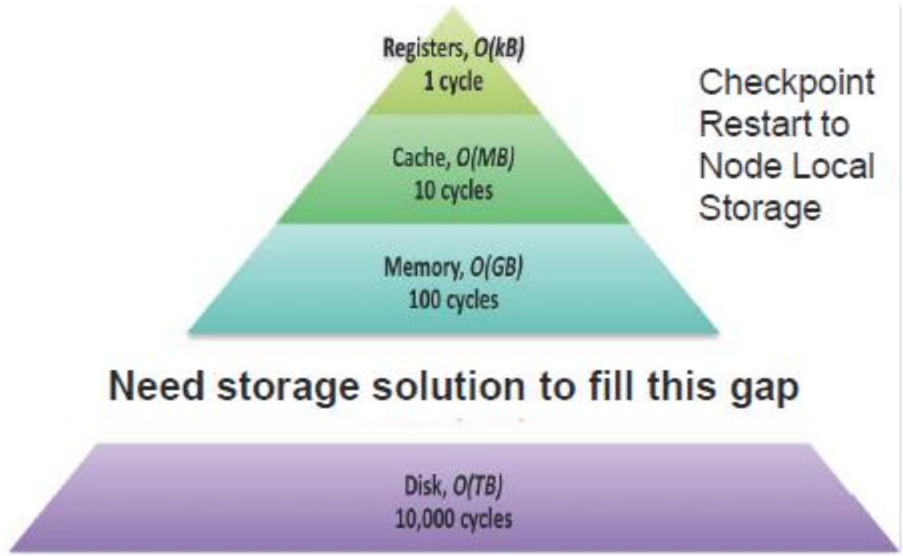
4) Co-design of burst buffers for visualization and analysis use cases



Burst Buffers Are Cost Effective Fast IO at Scale – Needed for Fault Tolerance

Must Meet Two Requirements: **900 PB Capacity and 100 TB/sec**

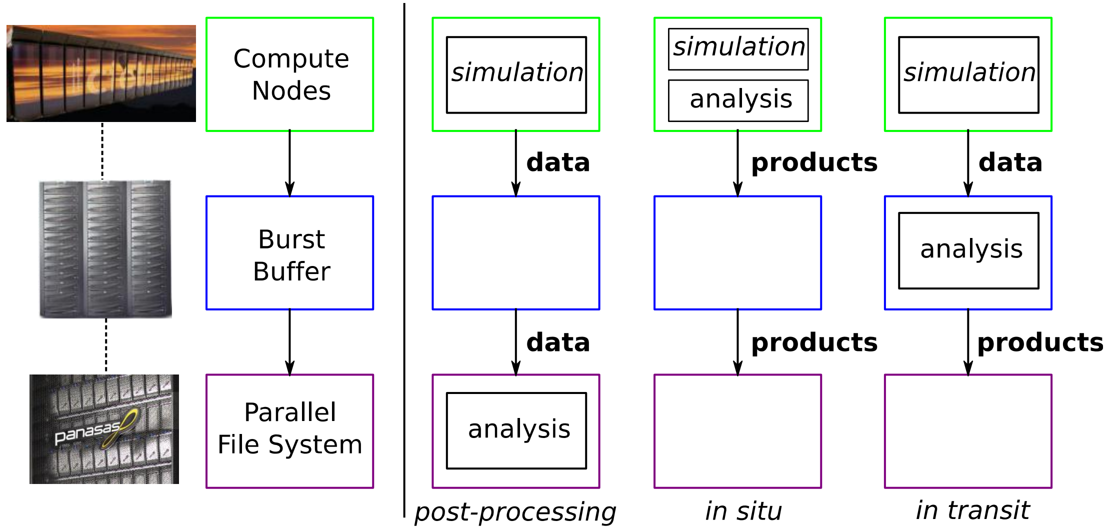




Mean time to failure shortens (more frequent) as supercomputers scale up. We use checkpoint restart to deal with failure, but it requires fast I/O, which can be expensive. Burst buffers are a cost effective solution for fast bandwidth…

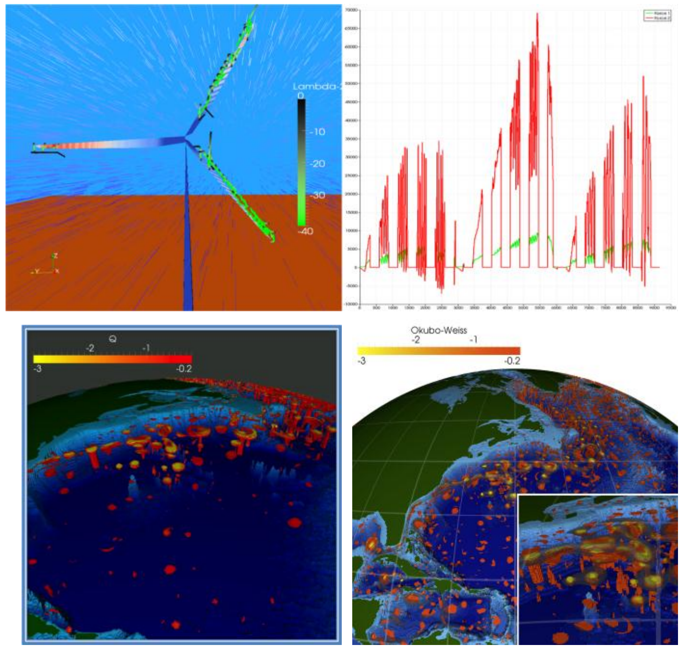
Can burst buffers be used for analysis?

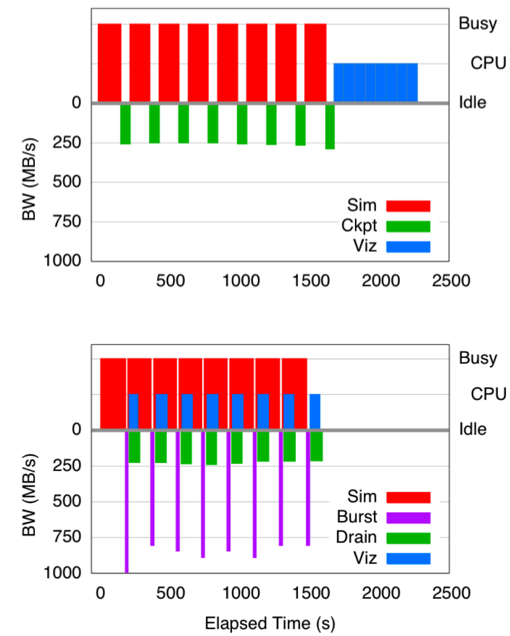
Hypothesis: Analysis Capability with Burst Buffers Optimizes Time to Results



- Different analyses are suitable for different stages in the pipeline – IO characteristics vs. compute
- Some are suited for post, some are suited for in situ, while others are suited for "in transit" on burst buffers

Supercomputing '11 and '12 demos show viability of "In Transit" analysis

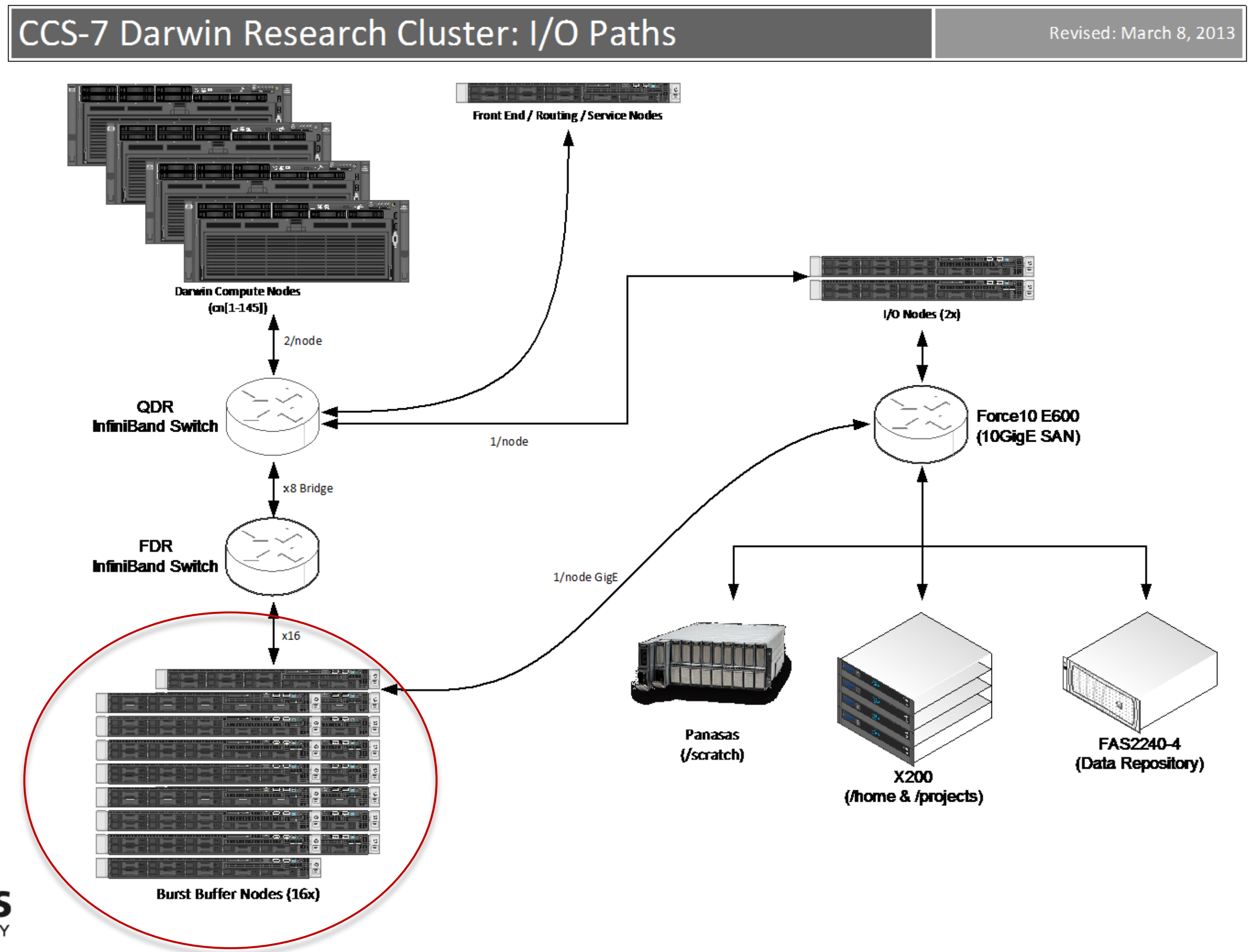




- HiGrad Firetec ('11) and POP ('12) ran live
- Faster turn around time to results due to pipeline parallelism of the burst buffers

Los Alamos
NATIONAL LABORATORY
EST.1943

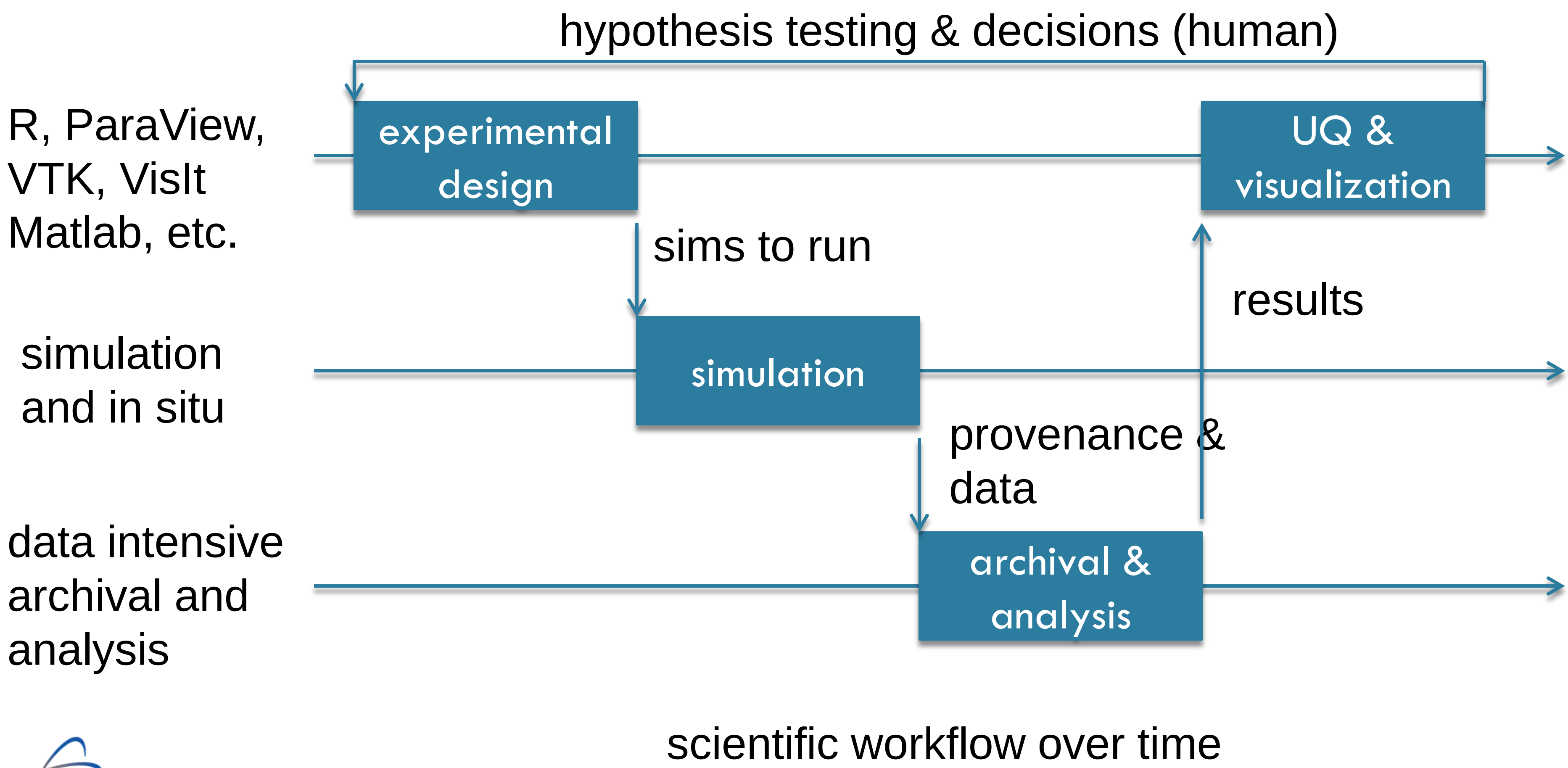
Hardware and Software Co-design with the sim and analysis use cases



Using POP (Parallel Ocean Program) as the use case – Different things to try

- Heavy I/O vs. heavy compute – MOC vs. Tracers
- Different software stacks to run on the burst buffer
  - Manage the burst buffer: I/O on buffers, analysis scheduling with simulation, data product creation
- Combine industry data intensive knowledge with HPC technologies and DOE analysis software
  - MPI with DISC? DISC with ParaView/VisIt/EnSight?
- How much faster can we turn around results for the scientist?
- Bring to bear all of the previous research I have discussed to create a prototype of the "vision"

Vision for HPC Scientific Workflow



Acknowledgements

ASC CSSE

ASCR Core and SciDAC

LANL LDRD

A Vision for Data Driven Science